**CHello**

In the era where teamwork and work collaboration are important, we need a system to manage, track, and organize everything to make sure it works well. **CHello** is a **Kanban-inspired** **collaborative work management application** that helps everyone with tracking and organizing everything whether personally or with team. With CHello, users get better visualization on a workflow and project control. It is currently under development, and you – as an expert in system analyst field – are assigned to **analyze** the system, **design** the system, and **develop** the application.

**1. Basic Concept of the Application**

* **Workspace**

CHello’s workspace is like a **virtual office**, where team members are located together, and team’s projects are **containerized** in one place. Basically, you can have **lots of projects** on a workspace and **different members** to work on them. Users – **main person** that will be **using** the **application** – are allowed to **create** as many workspaces as they want without any limitation.

* **Board**

In CHello, a project is called as ‘**board’**. This is where users **track** and **organize** everything they need; ongoing, pending, and completed tasks, team information, plans, resources, etc. Ideas and tasks that users have in a board are represented by ‘**card’**, and cards are organized in a ‘**list’**.

* **List**

Lists keep cards **organized** in their various **stages of process**. It can whether be used to create a **workflow** where cards are **moved** across each step in the process from start to finish or simply act as a place to keep **track** of **ideas** and **information**. A board can have as many lists as possible without any limitation and it **can be freely moved** on the board.

* **Card**

Card can be **customized** to hold a wide variety of useful **information**; titled, labeled, check listed, commented, and many more. A list can have as many cards as possible without any limitation, and card **can be freely moved** from a list to another.

**2. Application Requirements**

* **Users**

To be able to **access** CHello, users need to **have an account**. To have so, there must be a way for user to **register** themselves. You are free to implement what kind of **information** that users need to **fill in** for the registration process, as long as they are **reasonable**. User’s input during the registration process must be **validated** to make sure that there is no **invalid data** inside the **database**. With the registered account, users might **login** to the application. If user’s input **match** the **credentials** inside the database, **redirect** them to the **Home Page**.

Users are allowed to **change their settings** to **customize** their **preferences**. Users will be able to change their **personal information** such as display picture, biodata, password, etc. Make sure to add an **extra layer of security** when the user wants to change **confidential** **information**. Since CHello is a collaborative work application, users will **interact** with one another. Users will be able to set their **privacy settings**, so it is more comfortable for users who **don’t** want to be **bothered** with **notification** such as invitation, mentions, etc. Users can also set their **notification frequency**, whether the notification will be sent **instantly**, **periodically**, or **never**.

* **Home Page**

On the Home Page, users can **see** all their **joined** **workspaces**, **boards**, and **cards** that they are **involved in**. Boards that users mark as **favorite** must be easy to **navigate** to. To make it easier for users to **find** a workspace or a board, implement a **search bar** on the Home Page. Through the Home Page, users can easily **create** a workspace and board. Every workspace and board must have **sufficient information** to **differentiate** itself with one another, such as with name and description. When **creating a workspace**, users can choose to **invite** other users by inputting their **emails** or just **skip** the invitation process. If users **don’t** **have** – **or** **join** – **any workspace**, they **won’t be able to create a board**.

* **Workspace**

When users **create** a workspace, they **automatically** become the **admin of workspace**. Workspace’s admins are **allowed** to **manage** members of the workspace, such as **inviting** or **removing**. There are **two ways** to **invite** user; via **email invitation** or through **generated link**. To invite other users via email invitation, the process is no different from inviting users during workspace creation. Validate that those who **has been invited** or **join** the workspace **can’t receive the invitation email**. The invited users might rather **accept** or **reject** the invitation, and **all workspace’s member** will be **notified** when **new users join** the workspace. Workspace’s admin can also **generate a link** of the workspace, and anyone who **click** on the link can then **join the workspace** as long as they have a CHello’s account. You also need to set the **duration** for the link before it **expires** to ensure the safety of workspace. Once users **join** a workspace, they will be able to **view** other members, **view** and **edit** workspace’s content.

A workspace has its **visibility**, which can be freely managed by the workspace’s admin. Public workspaces can be **viewed** by **everyone**, including those who are not the workspace’s member, but only those who are **joined** can **add** or **edit** workspace. Private workspaces can only be **viewed** by their **member**. Workspace’s admin can also **grant**/**revoke** the **admin role** to workspace’s member. You are free to implement CHello’s **access control management**, but make sure to be **rational** and **reasonable** to protect the well-being of the workspace.

Workspace’s members are allowed to **leave** the workspace. If user is the only member of workspace, the workspace will **automatically** be **deleted**. Having **another user** as workspace’s admin **impacts** deletion process of the workspace. If there’s only **one admin** in the workspace, the workspace will be **automatically deleted** when the admin initiates it. Otherwise, the admin must **ask** for **other admins’ permission**. Workspace’s admins will be **notified** and after they **approve** the deletion, workspace will be **deleted**. When workspaces are **deleted**, all the boards on the workspace will be **closed**. However, all the closed board can still be **reopened** by board’s admin. To do so, board must be **moved** to another workspace. If board’s admins don’t wish to reopen the board, they might also **permanently close** the board.

* **Board**

Boards can be created **by any member** of workspace. Just like workspaces, those who create boards will **automatically** be the board’s admin. Boards have **three types of visibility**. A public board can be **viewed by everyone**, including those who are not a board’s member. A workspace-visible board can be **viewed by workspace’s member**. A board-visible board can only be **viewed by board’s member**. On every type of visibility, only board’s members are **allowed** to do **editing**.

Board’s admins have the **privilege** to **manage members** of the board, such as **inviting**, **removing**, and **grant/revoke** board’s admin role to board member. You can join a board via board’s admin **invitation email** or by **clicking** the **generated invitation link**. Make sure that board invitation process in CHello **doesn’t violate** its workspace’s **visibility** configuration. You need to **validate** that those who aren’t workspace’s member can only **join a public board**. Board’s members are also allowed to **leave** the board. If user is **the only member of board**, they need to choose wether to **close** the board or **permanently delete**.

After joining the board, users can **view** the **board details**. All board’s members have the **permission** to **create list** and **card** on the board. Users can also **search** and **filter** **shown lists** and **cards**. When the board grows, there might be lots of lists and cards on the board, so you need to implement **infinite scrolling** to load them. As a collaborative work application, users will work together on the board, make sure that **any change** in the board is **real time**.

Boards can either be **closed** or **permanently closed**. When board’s admins close a board, it can be **reopened**. Therefore, you need to **create a menu** where board’s admins can **view all closed board**. However, permanently closing a board **cannot be undone**.

Boards can also be viewed in **Calendar View**. Users can add a card based on **the date on the calendar**, and cards with **due date** will be **shown** on the Calendar **corresponding** to the date.

* **Card & List**

There are lots of things users can do with card. Users can basically **create**, **edit**, and **delete** a card. Remember that every card **must be contained** in a list, implement **drag and drop** to **move** card from a list to another. To **differentiate** lists and cards from one another, users might give **title** to lists and cards. List can also be **dragged and dropped** to **rearrange** them on the board. Most of the works that users do will be on cards. Cards can be added with components such as:

* + **Card’s Description**

Users can **add**, **edit**, or **remove** card’s description. Make sure to enable **text formatting** such as bold, italic, and underline to make it more convenient for users. As a desktop application, keyboard shortcuts can boost users productivity while using the app, so implements **keyboard shortcuts** to do the text formatting.

* + **Card’s Label**

With labels, users can **categorize** a card based on their **type**. Having labels on card will help users when **filtering**. Every label has its own **color** and **name**, and users can freely **create**, **edit**, or **remove** labels. These labels can be **attached** or **detached** from the card.

* + **Card’s Checklist**

Checklist can be used by users to **track** things that need to be done. Every item on the checklist can be **checked**/**unchecked**. **Display** a **progress bar** with **correct calculations** based on the checked items. Users are allowed to **create more than one** checklist in a card.

* + **Card’s Attachment**

Users must be able to **attach** **clickable links** and **files** on card. The file will include **file** such as images, text file, or other types of files. User can select a file to be **uploaded** or **drag** **and** **drop** the file to attach it. The file attached must be **downloadable** by those who can view the card. Users might also **detach** the attached file from card.

* + **Card’s Due Date**

Users can set a **specific due date** and **time** for card. They might also **set reminders** before the due date. To make it convenient for users, create a **date picker** so they can easily pick a date. You are free to implement the **option of reminder**, such as 1 day before, 1 week before, etc. Users can **mark** cards that have due date as **completed**. Cards with due date that are not marked as completed after the due date has passed will be categorized as **overdue**. Due date will also be useful for users to **filter** cards based on the **due date category**.

* + **Card’s Watcher**

Board’s member can be **assigned** as **watcher** for card by board’s admin. A card creator will **automatically** be a watcher. Watchers will get **notified** when cards meet their **due date** or **reminder date**. Remember **not** to violate their **privacy settings** while doing so.

* + **Card’s Location**

Users can **add** **location** on a card. When a location is added to card, **show** the location of the card with the **right coordinate** from the **map**.

* + **Card’s Link**

Just like workspace and board, card can be accessed through a **generated link**. For boards with board-visible type visibility, those who **aren’t** board’s member are **prohibited** to view the card, even if they have the card’s link.

* + **Card’s Comment**

Users might also give **comment** or **reply** to comments on cards. Card’s watchers will be **notified** when there is new comment. When commenting, users can also **mention** another user. **Validate** that if board is a **workspace’s visible board**, the users that can be mentioned are those who are workspace’s member. If board is a **board’s visible board**, users that can be mentioned are only board’s members.

* **Desktop Application Features**

Since CHello is a desktop application, you need to **utilize** the features to make it convenient to use. **Windows taskbar** is a component that will always be shown to the users, and users will interact with it the most. Use an **icon** to make the app interesting, create a **jump list** to list file, folder, or function that the user can access when **right clicking** on the application icon on the taskbar. Through the jump list, user can **navigate to their favorite boards**. Users must also be able to use the app in **full screen**. You might also implement **other keyboard shortcuts** than text formatting.

**3. IMPORTANT NOTES**

* You must **create** the **analysis diagrams** as stated below:
  + Use Case Diagram
  + Full Use Case Description
  + Activity Diagram
  + Class Diagram (Updated Class Diagram with relationship)
  + Multi-layer Sequence Diagram
* For Full Use Case Description, Activity Diagram, and Multi-layer Sequence Diagram, create **5 diagrams of each type** from 5 chosen use case. You are **not** allowed to choose use cases with simple logics, feel free to **consult** to the casemaker if you are not sure about your picks.
* You are allowed to make your own assumption of the diagram and concept, but it must be logical, and you must be able to explain all the diagrams that you made.
* All diagrams must be based on **Satzinger’s Design concepts**.
* Since the object-oriented approach presented in Satzinger’s Design concept is based on the **Unified Modeling Language (UML 2.0)** from the Object Management Group, as long as you **understand** the **concept**, you are allowed to create the diagrams based on it too.
* Your **application** must be made based on **your analysis diagrams**
* You must make the application using **Electron** based on **JS** / **TS** and **Firebase** backend
* You are **not** required to copy the original application, be creative on doing the user interface, but make sure that you **meet all the requirements** and consider the **user experience**.
* Your application must have at least **three design patterns**.